DOI 10.31891/2307-5732-2024-341-5-26
УДК 004.04

ДАВЛЕТОВА АЛІНА
Західноукраїнський національний університет
https://orcid.org/0000-0002-1192-2532
e-mail: a7davletova@gmail.com

АЛГОРИТМ ШИФРУВАННЯ ДАНИХ З КОРЕКЦІЄЮ ПОМИЛОК

В роботі досліджено ефективність поєднання асиметричної криптосистеми з кодами для корекції помилок з метою підвищення ефективності та безпеки обміну та зберігання інформації. Запропонований алгоритм використовує шифрування RSA для захисту даних від несанкціонованого доступу та коди Хеммінга в скінченних полях Галуа GF(p) для виявлення та виправлення помилок, що виникають під час передачі. Отримані результати демонструють підвищення надійності передачі даних, зберігаючи при цьому характеристики традиційної криптосистеми. Розглянуто застосування запропонованого підходу для повідомлень різної довжини та визначено оптимальний розмір вихідних даних, для яких алгоритм є ефективним.

Ключові слова: асиметрична криптосистема, коди для корекції помилок, скінченні поля Галуа, надійність передачі даних.

DAVLETOVA ALINA
West Ukrainian National University

ERROR CORRECTION DATA ENCRYPTION ALGORITHM

The vulnerability of data during transmission and storage necessitates the creation of effective comprehensive solutions for information protection. Research on combining cryptographic methods with error-correcting codes is relevant for developing reliable, secure, and efficient information protection systems. One of the most widely used methods for protecting data from unauthorized access is the RSA algorithm. Its high resilience is based on the difficulty of factoring large numbers, making it a reliable choice for safeguarding confidential information in various fields. Error-correcting methods play a crucial role in protecting data from corruption during transmission. They help prevent loss or modification of information by providing an additional layer of protection. One effective error-correcting method is Hamming codes, due to their simplicity of implementation and error-correcting properties. The proposed algorithm combines RSA cryptographic protection with error detection and correction capabilities using Hamming codes in finite fields GF(p), creating a robust and secure data protection system for transmission. The use of asymmetric cryptographic systems ensures data authenticity, while error-correcting codes maintain data integrity and accuracy by correcting errors that may occur due to noise or other issues in communication channels. This approach allows for a multi-layered strategy to data protection and enhances the efficiency of data transmission. Research on the proposed algorithm for messages of various lengths has demonstrated its effectiveness in handling large volumes of data. The benefits gained from integrating RSA encryption and error-correcting codes in finite fields GF(p) outweigh the minor increase in computational complexity, especially when it contributes to improving the reliability and cryptographic strength of the system.

Keywords: asymmetric cryptosystems, error correction codes, finite Galois fields, data transmission reliability.

Постановка проблеми

Криптографія є важливим елементом сучасного цифрового світу, забезпечуючи захист інформації в процесі обміну. Одним із найбільш поширених алгоритмів шифрування є RSA, відомий ефективністю та здатністю забезпечувати високий рівень безпеки, гарантуючи конфіденційність, цілісність та автентичність даних. Зростання складності процесів обробки та обсягів даних зумовлюють появу нових викликів, серед яких ключовим є необхідність виявлення та виправлення спотворень, що можуть виникати під час передачі або зберігання інформації. Ця задача успішно вирішується завдяки застосуванню методів корекції помилок, зокрема кодів Хеммінга, які набули широкого застосування завдяки своїм властивостям та простоті реалізації. Дослідження алгоритмів, що поєднують криптографічний захист з кодами для корекції помилок націлені на вирішення актуального завдання покращення функцій шифрування та підвищення надійності криптосистем у захисті від випадкових або навмисних модифікацій даних.

Метою даної роботи є дослідження переваг та можливостей поєднання асиметричного алгоритму шифрування з методом корекції помилок для подальшого розвитку та покращенню криптографічних засобів захисту інформації.

Аналіз останніх джерел

Дослідження та розвиток алгоритмів шифрування є невід'ємною частиною забезпечення кібербезпеки. Вони відіграють ключову роль для побудови ефективних систем захисту інформації, сприяючи дотриманню принципів CIA, протидії загрозам та забезпеченню безпеки цифрових комунікацій [1]. Зі зростанням кіберзлочинності, поширенням хмарних технологій та розвитком квантових комп'ютерів, необхідність у надійних криптографічних методах стає ще більш очевидною та актуальною. У сучасній науці велике коло вчених займаються розробкою та дослідженням стандартних та модифікованих алгоритмів RSA [2] і їх різноманітних застосувань, як для хмарної безпеки [3], криптографії зображень [4], безпровідних мереж та комунікацій [5, 6], пристроїв Інтернету речей [7] так і для інших областей.

В роботі [2] досліджено властивості поліномів Чебишева, що визначають перспективи їх застосування для криптографічних інструментів. Запропоновано покращений алгоритм шифрування з відкритим ключем, що поєднує хаотичне відображення Чебишева та RSA - $CRPKC{-}Ki$. Введено альтернативні коефіцієнти множення Ki, вибір яких визначається розміром $\mathrm{Tr}(\mathrm{Td}(x))\mathrm{mod\ N} = \mathrm{Td}(\mathrm{Tr}(x))\mathrm{mod\ N}$, а правила вибору

значень є спільними між учасниками, що дозволяє підвищити безпеку. Запропоновано використання більш складних проміжних процесів на етапах генерації ключів та шифрування / дешифрування для досягнення вищої складності алгоритму, що робить його більш стійким до звичайних атак.

В роботі [3] проведено аналіз двох криптографічних алгоритмів AES та RSA за параметрами часової складності, обсягу пам'яті для зберігання ключів, використання обчислювальних ресурсів та енергоспоживання. Запропоновано гібридний процес шифрування який поєднує переваги обох криптографічних систем. Підкреслено важливість інтеграції асиметричних методів для забезпечення безпеки ключів, що робить гібридний метод найбільш оптимальним вибором для забезпечення захисту даних у хмарному середовищі.

В роботі [4] запропоновано підхід для забезпечення покращеного шифрування даних шляхом комбінування алгоритму RSA та стеганографії. Представлено багаторівневий процес шифрування та дешифрування, який включає ряд етапів: перемішування алфавітно-цифрової послідовності, техніку доповнення криптографічного повідомлення, перетворення ASCII та алгоритм RSA, з подальшим вбудовуванням зашифрованого тексту в зображення за допомогою стеганографії. Проведена оцінка продуктивності запропонованого методу результати якої демонструють забезпечення високого рівня безпеки та ефективності комбінованого використання криптографії та стеганографії для захисту даних.

В роботі [5] представлено метод шифрування під назвою TPRSA заснований на тридекартовій алгебрі та поліномах, як альтернативу поліноміальному RSA і модифікованому RSA. Запропоновано використання модифікованої математичної структури ключів шифрування та дешифрування для досягнення високого рівня безпеки. Зазначено переваги запропонованого методу у порівнянні з існуючими, зокрема здатність одночасного шифрування шести повідомлень, що робить його цінним інструментом для застосувань, де необхідно здійснювати обробку даних з різних джерел одночасно.

В роботі [6] розглянуто метод підвищення безпеки алгоритму RSA шляхом приховування ключа шифрування e та спільного модуля криптосистеми n. Запропоновано метод зміни значень, які передаються публічно, і відрізняються від фактично використовуваних в алгоритмі RSA. Це збільшує час, необхідний для зламу криптосистеми, що підвищує її стійкість до криптоаналізу. Для реалізації використано два різні алгоритми для приховування кожного з параметрів, випадковий вибір між якими здійснюється за допомогою генератора випадкових чисел. Важливим аспектом запропонованого методу є забезпечення правильної комунікації інформації про використаний алгоритм, що здійснюється через окремий канал або за допомогою безпечного каналу зв'язку для коректного дешифрування даних.

В роботі [7] представлено покращену порогову схему агрегованих підписів на основі RSA для зменшення розміру блокчейну. Запропонована модифікація базового алгоритму RSA для покращення безпеки шляхом використання однакового модуля N для всіх вузлів мережі. Розроблена архітектура *AGS-MR* забезпечує ефективне управління підписами у багатовузлових системах без необхідності підтвердження вузлами знання чи володіння секретним ключем, що покращує загальну продуктивність і безпеку мережі. Показано, що розмір кінцевого агрегованого підпису, незалежно від кількості вузлів у мережі, залишається рівним $O(k)$, де k – параметр безпеки.

Аналіз джерел показав, що основною метою шифрування даних є забезпечення безпечної комунікації через незахищені канали зв’язку. Тому дослідження і розвиток методів захисту інформації, що передається, які гарантують конфіденційність, цілісність і автентичність даних, навіть якщо канал зв’язку може бути вразливим до несанкціонованого доступу чи прослуховування є перспективним напрямком досліджень. Проте, для ефективного обміну даними, наявність шуму і перешкод у каналах зв'язку створює необхідність застосування алгоритмів виявлення і виправлення спотворень. Ці методи у поєднанні з криптографічним захистом дозволять забезпечити як безпеку, так і цілісність переданих даних.

Дослідження алгоритмів шифрування даних

Традиційні симетричні системи шифрування є ефективними для безпечного обміну інформацією, проте їх безпека залежить від секретності ключа. Якщо він стає відомим, конфіденційність втрачається. Тому передача ключа стає все більш складною, особливо на великі відстані та при частій їх зміні. Концепція запропонована Діффі та Хеллманом [8] дозволяє вирішити проблему обміну ключами, особливо в багатокористувацьких мережах.

Асиметричні криптосистеми забезпечують більшу безпеку та дозволяють уникнути проблем, пов'язаних з обміном ключами, що властиві симетричним системам. Розроблено широке коло асиметричних криптоалгоритмів, безпека яких базується на різних обчислювальних задачах, наприклад схема ElGamal [9], криптосистема McEliece [10], криптографія на основі еліптичних кривих ECC [11], на основі решітки [12] та ін. Проте перший повноцінний криптографічний алгоритм з відкритим ключем RSA [13] і на сьогодні залишається високоефективним алгоритмом шифрування, не зважаючи на появу нових криптографічних методів, а також потенційної загрози з боку квантових комп'ютерів [14]. Він широко підтримується бібліотеками та протоколами, що робить його універсальним рішенням для різних систем та дозволяє інтегрувати в програмні продукти з метою забезпечення сумісності з існуючими криптографічними стандартами.

Стійкість алгоритму RSA до широкого кола атак безпосередньо залежить від довжини ключів і складності факторизації чисел, які використовуються для їх створення. Тому у забезпеченні безпеки алгоритму ключовим етапом є процес генерація ключів, що гарантує високий рівень випадковості та непередбачуваності, що включає наступні кроки [13]:

1. Генерацію простих чисел: вибір двох великих простих числа p та q.

2. Обчислення модуля $n = p\square\ q$, що є частиною відкритого та закритого ключів.

3. Обчислення функції Ейлера *φ(n)* де *φ(n)=(p-1)(q-1)* враховує кількість чисел, менших за *n* і взаємопростих з *n*.

4. Вибір відкритого ключа *e* таким, щоб $1 < e < (n)$ і найбільший спільний дільник НСД *(e, φ(n))* = 1.

5. Обчислення закритого ключа *d*, як мультиплікативно оберненого до *e* за модулем *φ(n)*. Тобто *d* задовольняє умову

$$e \cdot d \equiv 1 \, mod \, n, \tag{1}$$

що можна записати як $d = e^{-1} \, mod \, \varphi(n)$.

В результаті виконання даних кроків ключі матимуть вигляд: публічний ключ (e, n) - для шифрування або перевірки підпису та приватний ключ (d, n) - для розшифрування або створення цифрового підпису.

Для шифрування повідомлення *m*, де *m<n*, обчислюється

$$c = m^e \, mod \, n, \tag{2}$$

де *c* - зашифроване повідомлення.

Для дешифрування зашифрованого повідомлення *c*, обчислюється

$$m = c^d \, mod \, n. \tag{3}$$

Алгоритм шифрування RSA забезпечує конфіденційність і автентичність даних, використовуючи складні математичні операції, що ґрунтуються на принципах теорії чисел. Ці операції важко виконувати в зворотному порядку без знання приватного ключа, що забезпечує стійкість алгоритму до атак. RSA дозволяє лише авторизованому власнику приватного ключа дешифрувати повідомлення та підтверджувати автентичність підписів, забезпечуючи надійний обмін інформацією.

Дослідження алгоритмів корекції помилок

Дослідження методів корекції помилок [15-18] є актуальними, особливо в умовах нестабільних та вразливих каналів зв'язку. Їх використання дозволяє забезпечити цілісність і точність переданої інформації, що допомагає уникнути необхідності повторної передачі даних, підвищуючи ефективність каналів зв'язку.

У сучасних системах передачі та зберігання даних вибір відповідних коригуючих кодів відіграє ключову роль для забезпечення захисту даних від спотворення. Циклічні коди [19] є потужним інструментом у виявленні та виправленні помилок, проте вони характеризуються складними алгоритмами реалізації та вимагають значних обчислювальних ресурсів. BCH коди [20] дозволяють виявляти та корегувати більшу кількість помилок порівняно з кодами Хеммінга [21], але мають значно вищу обчислювальну складність. Вони застосовуються для критичних додатків, де забезпечення надійності є пріоритетною, але не завжди є оптимальними для систем з обмеженими ресурсами. Коди Ріда-Соломона [22] використовуються в багатьох високонадійних системах опрацювання даних, однак вони характеризуються складними алгоритми реалізації, що робить їх менш ефективними для ресурсо-обмежених систем.

Серед різноманітних варіантів, особливої уваги заслуговують коди Хеммінга [21], що базуються на принципі додавання контрольних бітів до повідомлення для виявлення і корекції помилок. Низька обчислювальна складність алгоритму корекції помилок дозволяє ефективно виявляти множинні та виправляти одиночні спотворення під час передачі даних, що робить їх оптимальними для систем з обмеженими ресурсами або необхідності швидкої обробки даних.

Коди Хеммінга, що використовуються в класичній теорії кодування, можуть бути адаптовані для побудови в скінченних полях Галуа *GF(p)*. Така модифікація забезпечує:

- розширення алфавіту - у класичній реалізації кожен символ коду є бінарним, у запропонованому випадку кожен символ може приймати значення з набору *{0, 1, ..., p−1}*.

- процес кодування та декодування залишається схожим, проте використовуються операції в *GF(p)*.

- коригуючи коди здатні виявляти та виправляти одну помилку в кожному блоці коду, де символи можуть приймати значення від *0* до *p−1*.

Алгоритм шифрування з корекцією помилок

Алгоритми шифрування в поєднанні з властивостями кодів для корекції помилок дозволяють забезпечити надійність передачі даних. Використання криптографічної системи з відкритим ключем забезпечує автентичність та безпеку даних, тоді як коригуючі коди зберігають їх цілісність. Це гарантує, що інформація не лише захищена від несанкціонованого доступу, але й залишається незмінною під час передачі.

Запропонований алгоритм створює багаторівневий підхід до безпеки інформації шляхом поєднання криптосистеми RSA з кодами Хеммінга в скінченних полях Галуа *GF(p)*. RSA шифрує дані для захисту, тоді як коригуючі коди виправляють помилки без значних затримок у передачі даних, що є критичним для систем реального часу. Інтеграція алгоритму шифрування RSA і кодів для корекції помилок побудованих в *GF(p)* можлива завдяки використанню спільного параметру *n* для виконання операцій, забезпечуючи таким чином комплексний підхід до захисту та обробки інформації.

На рисунку 1 наведено порядок виконання операцій запропонованого алгоритму: вихідне повідомлення кодується, а далі відбувається процес шифрування отриманого кодового слова. При цьому контрольні символи будуть зашифровані разом із інформаційними, що забезпечує додатковий захист від зловмисного втручання.

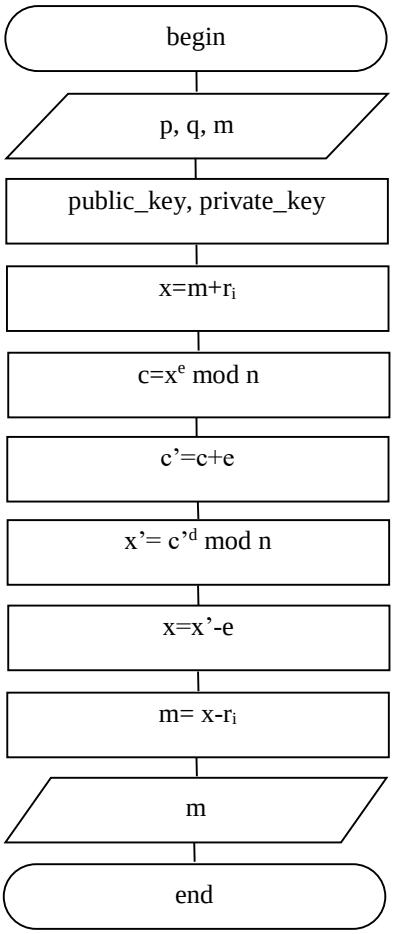

Рис. 1. Блок-схема роботи запропонованого алгоритму

Базовими операціями запропонованого алгоритму шифрування з корекцією помилок є:

1. Кодування: повідомлення *m* кодується в кодове слово *x* шляхом додавання контрольних символів r_i у позиції 2^i повідомлення для виявлення та виправлення однієї помилки. Кількість r_i повинна задовільняти умову:

$$2^r > k + r + 1, \tag{4}$$

де *k* – кількість інформаційних символів.

Обчислення контрольних символів, як суми відповідних інформаційних символів за модулем *n*

$$r_i = \sum k_i \; mod \; n. \tag{5}$$

Наприклад, для коду Хеммінга (15, 11) визначення значень r_i має вигляд:

$$\begin{gathered} r_0 = (k_3 + k_5 + k_7 + k_9 + k_{11} + k_{13} + k_{15}) mod \; n; \\ r_1 = (k_3 + k_6 + k_7 + k_{10} + k_{11} + k_{14} + k_{15}) mod \; n; \\ r_2 = (k_5 + k_6 + k_7 + k_{12} + k_{13} + k_{14} + k_{15}) mod \; n; \\ r_3 = (k_9 + k_{10} + k_{11} + k_{12} + k_{13} + k_{14} + k_{15}) mod \; n. \end{gathered} \tag{6}$$

2. Шифрування: вихідне повідомлення *x* перетворюється на *c* з використанням відкритого ключа (*e* ,*n*) RSA (2), де *e* – відкритий експонент, при чому задовольняється умова (1).

3. Передача зашифрованого повідомлення *c* каналами зв'язку, яка може супроводжуватися додаванням шумів, що призводить до виникнення помилки в переданих даних. Іноді для підвищення стійкості та надійності передачі інформації до повідомлення навмисно додається вектор помилки *e*

$$c' = c + e. \tag{7}$$

4. Дешифрування отриманого повідомлення *c'* за допомогою приватного ключа RSA (*d, n)*. Для кожного інформаційного символу повідомлення виконується (3).

5. Декодування кодового слова *x'* з потенційним помилковим символом d'_i з використанням алгоритму виявлення та виправлення помилок. Для визначення розміру помилки використаємо вираз

$$v_i = r_i - (\sum k_i - k'_i) \; mod \; n. \tag{8}$$

У контексті запропонованого способу, кожне число зашифрованого повідомлення є блоком, оскільки RSA, як правило, працює з блоками даних, що можуть бути перетворені на числа. Кожне з таких чисел представляє собою окремий інформаційний символ, який може бути зашифровано.

Запропонований алгоритм дозволяє використовувати переваги шифрування RSA без додаткових кроків при дешифруванні та забезпечує більш високий рівень безпеки, оскільки дані захищені від змін або модифікацій. Під час дешифрування не потрібно враховувати наявність контрольних символів, щоб

забезпечити правильну їх обробку. Проте, у випадку спотворення даних, застосування кодів Хеммінга в *GF(p)* дозволяє їх виявити і виправити, що забезпечує надійність і точність передачі.

Для реалізації запропонованого алгоритму використаємо набір модулів p=5, q=13, їх добуток складе n=5·13=65, та значення функції Ейлера, відповідно, $\varphi(n)$=48.

1. Кодування повідомлення. Перетворимо вихідне повідомлення m = 23,19,1,16,43,61,32,48,2,54,36 на кодове слово x в полі *GF(n)*. Підставивши відповідні значення інформаційних символів k_i в (6) отримаємо значення контрольних символів r_i:

$r_0 = (23 + 19 + 16 + 43 + 32 + 2 + 36) mod\ 65 = 41;$

$r_1 = (23 + 1 + 16 + 61 + 32 + 54 + 36) mod\ 65 = 28;$

$r_2 = (19 + 1 + 16 + 48 + 2 + 54 + 36) mod\ 65 = 46;$

$r_3 = (43 + 61 + 32 + 48 + 2 + 54 + 36) mod\ 65 = 16.$

Додамо отримані значення на позиції 2^i отримаємо кодове слово x:

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}
41	28	23	46	19	1	16	16	43	61	32	48	2	54	36

2. Шифрування. Кожен символ отриманого кодового слова x_i шифруємо використовуючи відкритий ключ (e, n) = (5, 65) відповідно до (2).

$$c_i = x_i^{\,e} mod\ n.$$

В результаті отримаємо зашифроване повідомлення c:

c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}	c_{11}	c_{12}	c_{13}	c_{14}	c_{15}
6	58	43	11	54	1	61	61	23	16	2	3	32	19	56

3. Передача даних. Припустимо, що в процесі передачі даних виникли перешкоди і до повідомлення c було додано випадкову помилку, вектор якої e = 0 0 0 0 0 0 0 0 0 0 20 0 0 0 0.

В результаті на приймальній стороні отримаємо спотворене повідомлення c':

c'_1	c'_2	c'_3	c'_4	c'_5	c'_6	c'_7	c'_8	c'_9	c'_{10}	c'_{11}	c'_{12}	c'_{13}	c'_{14}	c'_{15}
6	58	43	11	54	1	61	61	23	16	**22**	3	32	19	56

4. Дешифрування кожного символу c'_i отриманого повідомлення відбувається з використанням приватного ключа (d, n)=(29, 65) відповідно до (3)

$$c_i = c'_i{}^{\,d} mod\ n.$$

В результаті отримуємо дешифроване повідомлення x', оскільки воно ще містить помилку.

x'_1	x'_2	x'_3	x'_4	x'_5	x'_6	x'_7	x'_8	x'_9	x'_{10}	x'_{11}	x'_{12}	x'_{13}	x'_{14}	x'_{15}
41	28	23	46	19	1	16	16	43	61	**42**	48	2	54	36

5. Корекція помилок. Декодер гарантовано виявить і виправить одну помилку в кожному блоці отриманого повідомлення. Для визначення позиції відповідно до (6), проводимо перерахунок контрольних символів r'_i та порівняння їх з отриманими значеннями r_i.

$r'_1 = (23 + 19 + 16 + 43 + 42 + 2 + 36) mod\ 65 = 51;$

$r'_2 = (23 + 1 + 16 + 61 + 42 + 54 + 36) mod\ 65 = 38;$

$r'_3 = (19 + 1 + 16 + 48 + 2 + 54 + 36) mod\ 65 = 46;$

$r'_4 = (43 + 61 + 42 + 48 + 2 + 54 + 36) mod\ 65 = 26.$

Отримані дані вказують на наявність помилки у позиції 1011, що відповідає k_7. Після визначення її розміру (8), проводимо корекцію. В результаті отримуємо x = 41,28,23,46,19,1,16,16,43,61,32,48,2,54,36.

6. Дешифрування. Контрольні символи додані під час кодування необхідні лише для виявлення та виправлення помилок. Після декодування їх можна відкинути, оскільки вони більше не потрібні. В результаті виконання алгоритму отримаємо вихідне повідомлення m = 23,19,1,16,43,61,32,48,2,54,36.

Комбінування методів криптографії та корекції помилок значно ускладнює доступ до корисної інформації. Оскільки контрольні символи, які були додані для корекції помилок, також потрапляють під захист криптографічного шифрування. Це ускладнює несанкціонований доступ до інформації, оскільки необхідно не лише дешифрувати дані, але і декодувати їх, для відновлення оригінального повідомлення.

Дослідження ефективності запропонованого алгоритму

Для оцінки ефективності запропонованого алгоритму шифрування даних з корекцією помилок в рамках даного дослідження використано вхідні повідомлення m довжиною від 128 до 4096 біт, де кожен символ представляється числом типу int, займаючи 32 біти. В даному контексті проведено оцінку залежності розміру вихідного повідомлення L від використаного алгоритму шифрування. Для кожного з повідомлень розглянуто два варіанти: алгоритм шифрування без захисту від помилок L_c та запропонований алгоритм шифрування з використанням додаткових контрольних символів L_x. На основі обчислень отримано результати, що наведені в таблиці 1.

З таблиці 1 видно, що для коротких повідомлень, наприклад, 128 біт, використання запропонованого алгоритму призводить до збільшення розміру шифрованого повідомлення на 3 символи, тобто 75%. Проте зі збільшенням розміру повідомлення, 4096 біт, ефективність алгоритму зростає, оскільки необхідне додаткове введення 8 контрольних символів, що становить 6% від загального обсягу повідомлення.

Таблиця 1

Дослідження довжини вихідного повідомлення при використанні різних алгоритмах шифрування

Вхідне повідомлення		Вихідне повідомлення без захисту від помилок		Вихідне повідомлення із корекцією помилок		Надлишковість	
біт	символи	біт	символи	біт	символи	біт	символи
128	4	128	4	224	7	96	3
256	8	256	8	384	12	128	4
512	16	512	16	672	21	160	5
1024	32	1024	32	1248	39	224	7
2048	64	2048	64	2304	72	224	7
4096	128	4096	128	4352	136	256	8

На рисунку 2 наведено графіки залежності розміру вихідного повідомлення c від вхідних даних k, в результаті шифрування без захисту від помилок L_c та з використанням запропонованого алгоритму L_x.

З рисунка 2 видно надлишковість запропонованого алгоритму шифрування даних з корекцією помилок, яка в даному контексті означає певну кількість символів, що додаються до шифрованого повідомлення для забезпечення можливості виявлення і виправлення помилок під час передачі або зберігання даних та може бути розглянута як компроміс між надійністю (можливістю виправлення помилок) і обсягом даних.

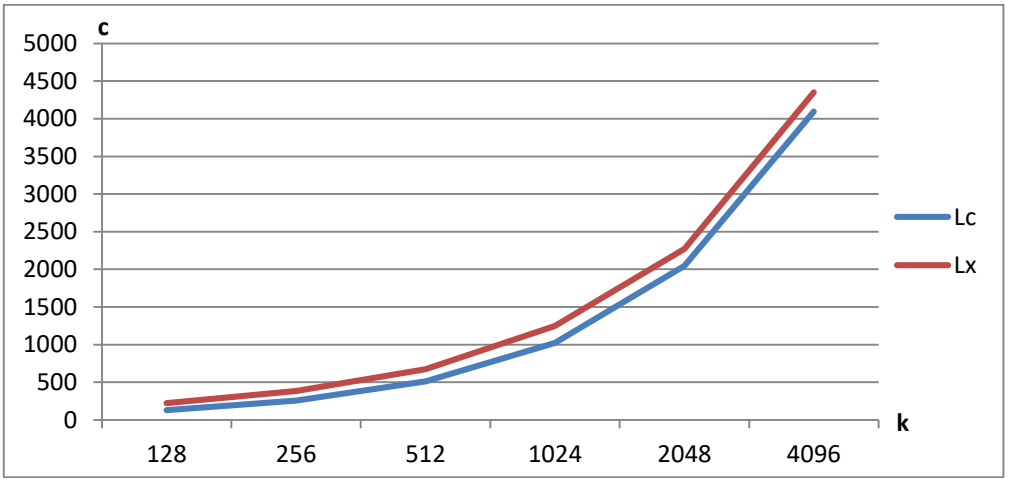

Рис. 2. Графік залежності довжини вихідного повідомлення від розміру вхідних даних

Таблиця 2 відображає обчислювальну складність основних етапів побудови коду для корекції помилок в скінченних полях Галуа, де k та x розмір вхідного та вихідного повідомлення відповідно.

Таблиця 2

Обчислювальна складність етапів побудови коду для корекції помилок

Назва етапу		Обчислювальна складність
Кодування:	Визначення кількості контрольних символів	$O(log\ k)$
	Ініціалізація контрольних символів	$O(x)$
	Обчислення значень контрольних символів	$O(x\ log\ x)$
Декодування:	Перевірка контрольних символів	$O(x\ log\ x)$.
	Виправлення помилки (за потреби)	$O(1)$

Обчислювальна складність для кодування і декодування повідомлень складе:

$$O(logk) + O(x) + O(x\ logx) + O(x\ log\ x) + O(1).$$

Зазвичай $x>k$, тому загальна обчислювальна складність кодування і декодування повідомлень за допомогою коду Хеммінга складе $O(x\ log\ x)$.

Обчислювальна складність алгоритму RSA залежить від реалізації кожного з його етапів, що наведена в таблиці 3, де K - кількість раундів тесту простоти, а N - розмір модуля.

Таблиця 3

Обчислювальна складність операцій алгоритму шифруння

Назва етапу		Обчислювальна складність
Генерація ключів	Генерація простих чисел p та q	$O(K \cdot log^3 N)$
	Обчислення n	$O(log^2 N)$
	Обчислення $\varphi(n)$	$O(log^2 N)$
	Вибір публічного експонента e	$O(1)$
	Обчислення приватного ключа d	$O(log^2 N)$
Шифрування		$O(log^2 N)$
Дешифрування		$O(log^3 N)$

Обчислювальна складність алгоритму шифрування RSA може бути виражена як сума всіх операцій:

$$O(K \cdot log^3 N) + O(log^2 N) + O(log^2 N) + O(1) + O(log^2 N) + O(log^2 N) + O(log^3 N).$$

Етап генерації ключів вимагає складних обчислень, оскільки включає вибір великих простих чисел та обчислення модульної оберненої. Шифрування є відносно швидким процесом завдяки використанню стандартного значення e, тоді як дешифрування вимагає більш ресурсів через розмір приватного ключа d. Оскільки даний етап домінуючий з точки зору складності, тому загальна обчислювальна складність алгоритму RSA визначається як $O(log^3 N)$. Як правило довжина модуля N пропорційна довжині блоку, який шифрується n, тобто $\log N \approx$ n, тому обчислювальну складність етапу дешифрування можна спростити до $O(n^3)$.

Визначити загальну обчислювальну складність запропонованого алгоритму можна як суму складностей всіх його кроків. Однак, для практичних цілей, зосередимося на домінуючому етапі - дешифрування, оскільки він суттєво впливає на загальні витрати обчислювальних ресурсів, а кодування та декодування є менш ресурсозатратними. Оскільки обчислення включають необхідність обробки додаткових контрольних символів, тому обчислювальна складність алгоритму шифрування з корекцією помилок визначається як $O((n + r)^3)$, де r - кількість контрольних символів. Результати дослідження обчислювальної складності запропонованого алгоритму шифрування для повідомлень різної довжини наведено в таблиці 4.

Таблиця 4

Дослідження обчислювальної складності

Вхідні дані m, біт	Обчислювальна складність		Зростання обчислювальної складності, %
	шифрування без корекції помилок	шифрування з корекцією помилок	
16	4,10E+03	9,26E+03	126,10
32	3,28E+04	5,49E+04	67,46
64	2,62E+05	3,58E+05	36,53
128	2,10E+06	2,52E+06	19,95
256	1,68E+07	1,86E+07	10,92
512	1,34E+08	1,42E+08	5,97
1024	1,07E+09	1,11E+09	3,26
2048	8,59E+09	8,74E+09	1,77
4096	6,87E+10	6,94E+10	0,96

Отримані результати свідчать про те, що із збільшенням розміру повідомлення додаткова складність обчислень, яку вносить алгоритм корекції помилок, зменшується на фоні загального експоненційного зростання складності обчислень. На рисунку 3 наведено графік зміни обчислювальної складності у відсотках залежно від розміру вхідних даних.

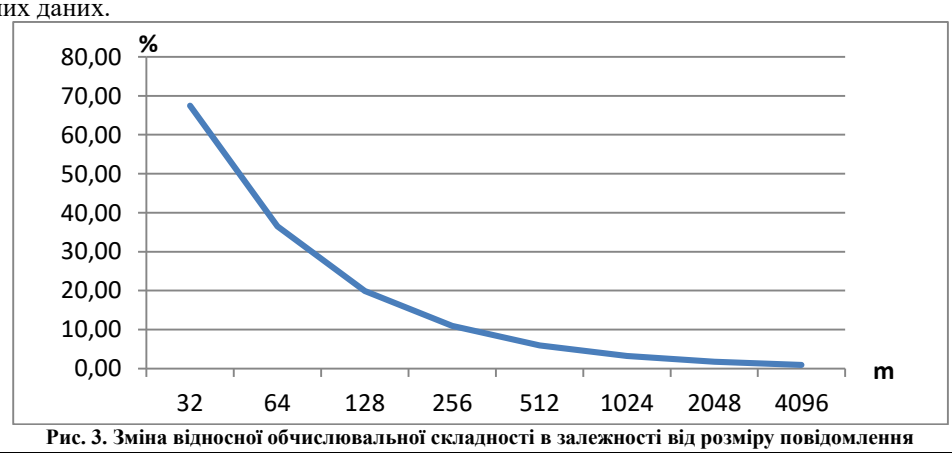

Рис. 3. Зміна відносної обчислювальної складності в залежності від розміру повідомлення

З рисунка 3 видно, що зі збільшенням обсягу вхідних даних відносне зростання обчислювальної складності запропонованого алгоритму знижується. Шифрування коротких повідомлень 32 біти, призводить до значного збільшення обчислювальної складності до 69,5%, тоді як для повідомлень 4096 біт вплив додаткових витрат на загальну обчислювальну складність алгоритму є менш значущим і становить 0,96%.

Висновки

Отримані результати демонструють, що інтеграція алгоритму корекції помилок у процес шифрування дозволяє підвищити рівень захисту даних, що є важливим в умовах обміну даними через нестабільні або вразливі канали зв'язку. Запропонований алгоритм забезпечує підвищення надійності передачі даних за рахунок використання властивостей кодів для корекції помилок побудованих у скінченних полях Галуа *GF(p)*.

Результати досліджень показали, що запропонований алгоритм є більш ефективним для шифрування великих обсягів даних. При розмірі вхідних даних 128 біт надлишковість складає 75% , а при збільшенні їх обсягу до 4096 біт - 6,25% від переданої інформації, що є обґрунтованим з урахуванням підвищення надійності. Відносне зростання обчислювальної складності запропонованого алгоритму знижується при збільшенні довжини повідомлення. Для повідомлень довжиною 32 біт, додавання символів для корекції помилок призводить до збільшення обчислювальної складності на 69,5%. Для даних обсягом 4096 біт обчислювальна складність запропонованого алгоритму наближається до складності алгоритму шифрування без корекції помилок, із збільшенням лише на 0,96%. Це свідчить про те, що для великих обсягів даних підвищення надійності може бути досягнуто без значного збільшення обчислювальних витрат.

Підвищення рівня захисту інформації забезпечується також шляхом шифрування контрольних символів, що забезпечує додатковий рівень безпеки, ускладнюючи несанкціонований доступ і модифікацію даних. Незважаючи на збільшення обчислювального навантаження, для генерації та обробки контрольних символів, запропонований підхід може бути ефективним з точки зору загальної надійності і безпеки передачі великих обсягів даних.

Література

1. Kapalova N., Sakan K., Algazy K., Dyusenbayev D. Development and Study of an Encryption Algorithm. Computation. 2022; 10(11):198. https://doi.org/10.3390/computation10110198

2. Zhang C., Liang Y., Tavares A., Wang L., Gomes T., Pinto S. An Improved Public Key Cryptographic Algorithm Based on Chebyshev Polynomials and RSA. Symmetry. 2024. 16, 263. https://doi.org/10.3390/sym16030263

3. Fatima S., Rehman T., Fatima M., Khan S., Ali M.A. Comparative Analysis of Aes and Rsa Algorithms for Data Security in Cloud Computing. Engineering Proceedings. 2022. 20(1):14. https://doi.org/10.3390/engproc2022020014

4. Karthikeyan B., Bharathkumar B., Manikandan G., Seethalakshmi R. A Combination of RSA Algorithm with Image Steganography to Ensure Enhanced Encryption. 2023 Second International Conference on Electronics and Renewable Systems (ICEARS), Tuticorin, India. 2023. p. 773-777. doi: 10.1109/ICEARS56392.2023.10085371.

5. Al-khuzaie B., Yassein H. Design of an Alternative to Polynomial Modified RSA Algorithm. Design of an Alternative to Polynomial Modified RSA Algorithm. International Journal of Mathematics and Computer Science. 19. 693-696. International Journal of Mathematics and Computer Science. 2024. 19. 693-696.

6. Balasubramanian K., Arun M., Sekar K.R. An Improved RSA Algorithm for Enhanced Security. Indian Journal of Cryptography and Network Security. 2022. 2. 1-4. https://doi.org/10.54105/ijcns.B1421.112222. https://www.researchgate.net/publication/365874002_An_Improved_RSA_Algorithm_for_Enhanced_Security

7. Chait K., Laouid A., Kara M., Hammoudeh M., Aldabbas O., Al-Essa A.T. An Enhanced RSA-Based Aggregate Signature Scheme to Reduce Blockchain Size. IEEE Access. 2023. 11. p. 110490–110501.

8. Hellman M. New directions in cryptography. IEEE transactions on Information Theory. 1976. T. 22. № 6. p. 644-654.

9. Elgamal T. A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Transactions on Information Theory. 1985. Vol. 31, no. 40. p. 469-472. https://doi.org/10.1109/TIT.1985.1057074.

10. McEliece Robert J. A public key cryptosystem based on algebraic coding theory. DSN Progress Report. 1978. p. 42-44.

11. Yakymenko I., Kasianchuk M., Yatskiv V., Shevchuk R., Koval V., Yatskiv S. Sustainability and Time Complexity Estimation of Cryptographic Algorithms Main Operations on Elliptic Curves. 11th International Conference on Advanced Computer Information Technologies (ACIT). 2021. p. 494-498. https://doi.org/10.1109/ACIT52158.2021.9548534.

12. Howe J., Poppelmann T., O'Neill M., O'Sullivan E., Guneysu T. Practical Lattice-Based Digital Signature Schemes. ACM Transactions on Embedded Computing Systems. 2015. 14(3). Article 41. https://doi.org/10.1145/2724713

13. Rivest R.L., Shamir A., Adleman L.M. A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM. 1978. Vol. 21, No. 2. p. 120-126.

14. Gigovic B. Breaking Encryption Algorithms: Methods, Tools, and Practical Examples. https://medium.com/@boris.gigovic/breaking-encryption-algorithms-methods-tools-and-practical-examples-

1e843244eead

15. Semerenko V. On the error-correcting capabilities of iterative error correction codes. Eastern-European Journal of Enterprise Technologies. 2019. 1(4). p. 31–39. https://doi.org/10.15587/1729-4061.2019.157299

16. Chi H., Liu J.-C., Chang C.-C., Horng J.-H. A Simple and Efficient Data Hiding Method with Error Detection and Correction. Electronics. 2024. 13(11):2018. https://doi.org/10.3390/electronics13112018

17. He Y., Wang Y. Research on Low-Density Parity-Check Decoding Algorithm for Reliable Transmission in Satellite Communications. Applied Sciences. 2024. 14(2):746. https://doi.org/10.3390/app14020746

18. Wang L.-N., Wei H., Zheng Y., Dong J., Zhong G. Deep Error-Correcting Output Codes. Algorithms. 2023. 6(12):555. https://doi.org/10.3390/a16120555

19. Різник В.В., Скрибайло-Леськів Д.Ю., Бадзь В.М. Порівняльний аналіз ефективності монолітного та циклічного завадостійких кодів. Український журнал інформаційних технологій. 20210. т. 3, № 1. с. 99–105.

20. Krylova V.A., Tverytnykova E.E., Vasylchenkov O.G., Kolisnyk T.P. Modified algorithm for searching the roots of the error locators polynominal while decoding BCH codes. Radio Electronics, Computer Science, Control. 2020. № 3. p. 150–157. ttps://doi.org/10.15588/1607-3274-2020-3-14.

21. Денбновецький С.В., Мельник І.В., Писаренко Л.Д. Кодування сигналів в електронних системах. Частина 3. Способи кодування сигналів. Том 1. Натуральні, ефективні та лінійні коди. Київ: КПІ ім. Ігоря Сікорського. 2021. 470 с.

22. Ваврук Є.Я., Попович Б.Р., Попович Р.Б. Програмна модель кодів Ріда-Соломона. Computer Systems And Networks. 2021. Vol. 3, No. 1. p. 1–6. https://doi.org/10.23939/csn2021.01.001

References

1. Kapalova N., Sakan K., Algazy K., Dyusenbayev D. Development and Study of an Encryption Algorithm. Computation. 2022; 10(11):198. https://doi.org/10.3390/computation10110198

2. Zhang C., Liang Y., Tavares A., Wang L., Gomes T., Pinto S. An Improved Public Key Cryptographic Algorithm Based on Chebyshev Polynomials and RSA. Symmetry. 2024. 16, 263. https://doi.org/10.3390/sym16030263

3. Fatima S., Rehman T., Fatima M., Khan S., Ali M.A. Comparative Analysis of Aes and Rsa Algorithms for Data Security in Cloud Computing. Engineering Proceedings. 2022. 20(1):14. https://doi.org/10.3390/engproc2022020014

4. Karthikeyan B., Bharathkumar B., Manikandan G., Seethalakshmi R. A Combination of RSA Algorithm with Image Steganography to Ensure Enhanced Encryption. 2023 Second International Conference on Electronics and Renewable Systems (ICEARS), Tuticorin, India. 2023. p. 773-777. doi: 10.1109/ICEARS56392.2023.10085371.

5. Al-khuzaie B., Yassein H. Design of an Alternative to Polynomial Modified RSA Algorithm. Design of an Alternative to Polynomial Modified RSA Algorithm. International Journal of Mathematics and Computer Science. 19. 693-696. International Journal of Mathematics and Computer Science. 2024. 19. 693-696.

6. Balasubramanian K., Arun M., Sekar K.R. An Improved RSA Algorithm for Enhanced Security. Indian Journal of Cryptography and Network Security. 2022. 2. 1-4. https://doi.org/10.54105/ijcns.B1421.112222. https://www.researchgate.net/publication/365874002_An_Improved_RSA_Algorithm_for_Enhanced_Security

7. Chait K., Laouid A., Kara M., Hammoudeh M., Aldabbas O., Al-Essa A.T. An Enhanced RSA-Based Aggregate Signature Scheme to Reduce Blockchain Size. IEEE Access. 2023. 11. p. 110490–110501.

8. Hellman M. New directions in cryptography. IEEE transactions on Information Theory. 1976. T. 22. № 6. p. 644-654.

9. Elgamal T. A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Transactions on Information Theory. 1985. Vol. 31, no. 40. p. 469-472. https://doi.org/10.1109/TIT.1985.1057074.

10. McEliece Robert J. A public key cryptosystem based on algebraic coding theory. DSN Progress Report. 1978. p. 42-44.

11. Yakymenko I., Kasianchuk M., Yatskiv V., Shevchuk R., Koval V., Yatskiv S. Sustainability and Time Complexity Estimation of Sryptographic Algorithms Main Operations on Elliptic Curves. 11th International Conference on Advanced Computer Information Technologies (ACIT). 2021. p. 494-498. https://doi.org/10.1109/ACIT52158.2021.9548534.

12. Howe J., Poppelmann T., ONeill M., OSullivan E., Guneysu T. Practical Lattice-Based Digital Signature Schemes. ACM Transactions on Embedded Computing Systems. 2015. 14(3). Article 41. https://doi.org/10.1145/2724713

13. Rivest R.L., Shamir A., Adleman L.M. A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM. 1978. Vol. 21, No. 2. p. 120-126.

14. Gigovic B. Breaking Encryption Algorithms: Methods, Tools, and Practical Examples. https://medium.com/@boris.gigovic/breaking-encryption-algorithms-methods-tools-and-practical-examples-1e843244eead

15. Semerenko V. On the error-correcting capabilities of iterative error correction codes. Eastern-European Journal of Enterprise Technologies. 2019. 1(4). p. 31–39. https://doi.org/10.15587/1729-4061.2019.157299

16. Chi H., Liu J.-C., Chang C.-C., Horng J.-H. A Simple and Efficient Data Hiding Method with Error Detection and Correction. Electronics. 2024. 13(11):2018. https://doi.org/10.3390/electronics13112018

17. He Y., Wang Y. Research on Low-Density Parity-Check Decoding Algorithm for Reliable Transmission in Satellite Communications. Applied Sciences. 2024. 14(2):746. https://doi.org/10.3390/app14020746

18. Wang L.-N., Wei H., Zheng Y., Dong J., Zhong G. Deep Error-Correcting Output Codes. Algorithms. 2023. 6(12):555. https://doi.org/10.3390/a16120555

19. Riznyk V.V., Skrybailo-Leskiv D.Iu., Badz V.M. Porivnialnyi analiz efektyvnosti monolitnoho ta tsyklichnoho zavadostiikykh kodiv. Ukrainskyi zhurnal informatsiinykh tekhnolohii. 20210. t. 3, № 1. c. 99–105.

20. Krylova V.A., Tverytnykova E.E., Vasylchenkov O.G., Kolisnyk T.P. Modified algorithm for searching the roots of the error locators polynominal while decoding BCH codes. Radio Electronics, Computer Science, Control. 2020. № 3. p. 150–157. ttps://doi.org/10.15588/1607-3274-2020-3-14.

21. Denbnovetskyi S.V., Melnyk I.V., Pysarenko L.D. Koduvannia syhnaliv v elektronnykh systemakh. Chastyna 3. Sposoby koduvannia syhnaliv. Tom 1. Naturalni, efektyvni ta liniini kody. Kyiv: KPI im. Ihoria Sikorskoho. 2021. 470 s.

22. Vavruk Ye.Ia., Popovych B.R., Popovych R.B. Prohramna model kodiv Rida-Solomona. Computer Systems And Networks. 2021. Vol. 3, No. 1. p. 1–6. https://doi.org/10.23939/csn2021.01.001